

PATENT ABSTRACTS OF JAPAN

22141 U.S. PTO

10/761377



012204

(11)Publication number : 2000-172521

(43)Date of publication of application : 23.06.2000

(51)Int.Cl.

G06F 11/18
G06F 15/177

(21)Application number : 10-351792

(71)Applicant : TOYOTA MOTOR CORP

(22)Date of filing : 10.12.1998

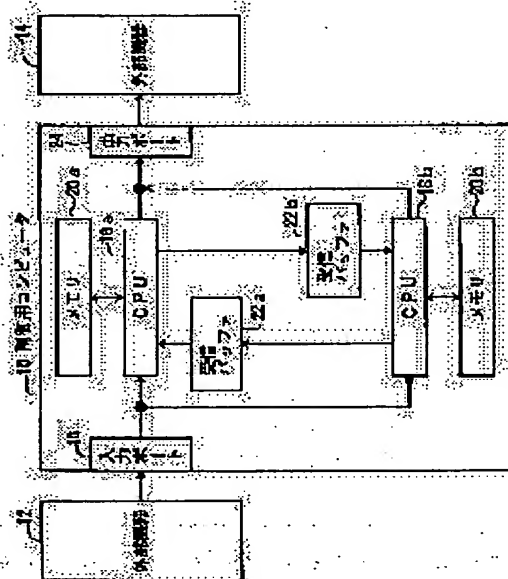
(72)Inventor : UCHIDA KIYOYUKI

(54) ABNORMALITY DETECTING METHOD FOR CPU

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a method for easily detecting the abnormality of a CPU without synchronizing a plurality of CPUs with each other.

SOLUTION: An input signal is supplied from an external device 12 to CPUs 18a and 18b through an input port 16. The CPUs 18a and 18b obtain the latent arithmetic values of the opposite CPUs through receiving buffers 22a and 22b and compares the arithmetic values with the latest three arithmetic values. Then the CPUs 18a and 18b judge that at least one of the CPUs 18a and 18b in a control computer 10 is abnormal when the latest arithmetic values of the opposite CPUs are larger than the maximum values among their arithmetic values or smaller than the minimum values. If the CPU 18a is judged to be abnormal, the output of a control signal supplied to the external device 14 is stopped. When it is judged that the CPU 18b is abnormal, on the other hand, the output of a control signal from the CPU 18a is inhibited.



LEGAL STATUS

[Date of request for examination]

29.08.2001

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

THIS PAGE BLANK (USPTO)

(18) 日本型特許庁 (J P) (12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-172521

(P2000-172521A)

(43) 公開日 平成12年6月23日 (2000.6.23)

(51) Int. Cl.	IPC	分類記号	PI	ターミナル(参考)
G06F 11/18	310	G06F 11/18	310C	5B034
15/177	678	15/177	678A	5B045

審査請求 未請求 請求項の数 2 OL (全 7 頁)

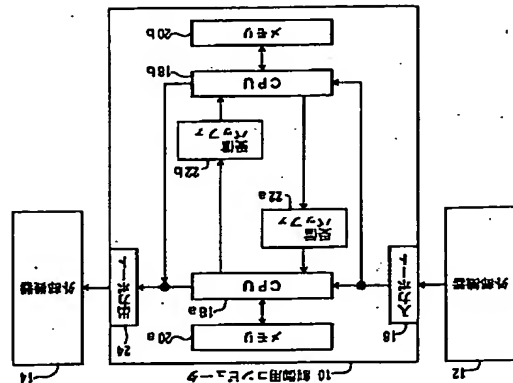
(21) 出願番号	特開平10-351792	(71) 出願人	00000207 トヨタ自動車株式会社
(22) 出願日	平成10年12月10日 (1998.12.10)	(72) 発明者	愛知県豊田市トヨタ町1番地 内田 清之
		(73) 発明者	愛知県豊田市トヨタ町1番地 トヨタ自動車株式会社
		(74) 代理人	100070150 弁理士 伊東 雄彦
		Fターム(参考)	5B034 A02 C01 D01 5B045 J103 J113

(54) 発明の名称 CPUの異常検出方法

(57) 【要約】

【課題】本発明は、複数のCPUの同期をとることなく、CPUの異常を容易に検出する方法を提供することを目的とする。

【解決手段】入力信号は、外部機器12から入力ポート18を介してCPU18a、18bに与えられる。CPU18a、18bは、受信バッファ22a、22bを介して相手側CPUの最新の演算値を取得し、該演算値を最新の3回の自己演算値と比較する。そして、CPU18a、18bは、相手側CPUの最新の演算値が自己演算値の最大値より大きい場合、又は、最小値より小さい場合に例外用コンピュータ10内のCPU18a、18bの少なくとも一方が異常であると判断する。CPU18aが異常ありと判断した場合は、外部機器14へ与える同期信号の出力を停止する。また、CPU18bが異常ありと判断した場合は、CPU18aからの同期信号の出力を禁止する。



【特許請求の範囲】

【請求項1】複数のCPUを有するコンピュータシステムにおけるCPUの異常検出方法であって、

各CPUにより所定の同期値を算出する第1のステップと、

各CPUにより他のCPUが算出した最新の演算値を取得する第2のステップと、

前記第2のステップの実行後に、各CPUによりそれぞれの入力信号に対する演算値を算出する第3のステップと、

各CPUが算出した最新の演算値と、他のCPUが算出した最新の演算値との大小関係の比較を互いに繰り返す第4のステップと、

前記比較の結果に基づきCPUの異常を検出する第5のステップとを備えることを特徴とするCPUの異常検出方法。

【請求項2】請求項1記載のCPUの異常検出方法であって、

前記第4のステップにおける最新の演算値は、最新の3つの演算値であることを特徴とするCPUの異常検出方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、CPUの異常検出方法に係り、特に、複数のCPUを有するコンピュータシステムにおけるCPUの異常検出方法に関する。

【0002】

【従来の技術】従来より、内部に複数のCPU (Central Processing Unit) を備えて動作信頼性を向上させたコンピュータシステムが広く知られている。このようなコンピュータシステムでは、複数のCPUの演算結果を比較することにより、CPUの故障等の有無が検出される。例えば、特開平5-324391号では、バス比較器によってCPUの故障の有無を検出する方法が開示されている。

【0003】このバス比較器は、複数のCPU毎に対応して設けられた圧縮処理部と比較器とを有している。そして、それぞれの圧縮処理部は、対応するCPUのビット毎のデータ圧縮部と直列伝送部を有している。このように構成において、CPUの出力データは、バス比較器内のデータ圧縮部によって符号圧縮される。符号圧縮されたデータは、直列伝送部によって直列に転送された後に比較器側に出力される。比較器は、各圧縮処理部から出力された圧縮データを順次比較して、不一致のデータがある場合にCPUの故障等を検出する。

【0004】このような故障検出方法によれば、CPUの出力データがデータ圧縮部によって圧縮されるので、CPUのバスのビット数が増加しても対応することができ、また、CPUの出力データが圧縮されて比較結果の出力間隔が低くなるので、フェールセーフを確保す

る分周回路等を設ける必要もなく、バス比較器等の低コスト化が図られる。

【0005】

【発明が解決しようとする課題】しかし、上記従来例のような方法でCPUの異常を検出するには、複数のCPUの演算タイミングや外部センサ等から複数のCPUに与えられるデータのタイミング等の同期をとる必要がある。このため、同期信号発生回路から各CPUに同期信号を与える必要があった。

【0006】また、CPUの同期をとるためにCPUが各処理毎に要する演算時間等を厳密に見積もる必要がある。更に、複数のCPUの同期をとるための同期信号がうまく働かず、CPUの同期がとれない場合の対策を準備しておく必要があった。上記従来例において、完全な同期がとられていない複数のCPUが演算を行なうとそれぞれの演算結果に差異が生じ、互いの演算結果の比較の結果、正常なCPUでも異常であると誤認識されたり、異常なCPUでも正常であると誤認識される可能性がある。

【0007】本発明は、上記の点に鑑みてなされたものであり、複数のCPUを有するコンピュータシステムにおいて、複数のCPUの同期をとることなく、CPUの異常を容易に検出する方法を提供することを目的とする。

【0008】

【課題を解決するための手段】上記目的は、請求項1に記載する如く、複数のCPUを有するコンピュータシステムにおけるCPUの異常検出方法であって、各CPUにより所定の同期値を算出する第1のステップと、各CPUにより他のCPUが算出した最新の演算値を取得する第2のステップと、前記第2のステップの実行後に、各CPUによりそれぞれの入力信号に対する演算値を算出する第3のステップと、各CPUが算出した最新の演算値と、他のCPUが算出した最新の演算値との大小関係の比較を互に行なう第4のステップと、前記比較の結果に基づきCPUの異常を検出する第5のステップとを備えるCPUの異常検出方法による達成される。

【0009】このようなCPUの異常検出方法では、複数のCPUが互いの演算値を比較し合うことにより、複数のCPUの同期をとることなくCPUの異常が容易に検出される。また、本発明は、他のCPUの最新の演算値を取得してから、各CPUが自己の演算値を算出した後、各CPUによって、取得した他のCPUの最新の演算値は、常に自己の最新の演算値よりも先に算出されたものとなる。このため、各CPUは、既に算出した最新の演算値と他のCPUが算出した最新の演算値との差に基づき、CPUの異常の有無を検出すればよく、自己の最新の演算値に依り算出した演算値を比較時に考慮する必要がない。従って、比較時の演算値を小さく設定することができ、高精度の演算の許容範囲を小さく設定することができ、高精度

なCPUの異常検出が実現する。

[0010] また、上記目的は、請求項2に記載する如く、請求項1記載のCPUの異常検出方法であって、前記第4のステップにおける最新の所定数の演算値は、最新の3つの演算値であるCPUの異常検出方法により達成される。各CPUが取得した他のCPUの最新の演算値は、各CPUが取得した他のCPUの最新の演算値は、自己の前回の演算処理時から最新の演算処理時の間に算出されたものであると限定できる。本発明によると、相手側CPUの最新の演算値と比較する自己CPUの演算値が最新の3つの演算値に限定されるので、比較時の演算の計算量をより小さな値に設定することができ、計算処理量をより小さな値に設定することで、CPUの異常の検出がより高精度に行なわれる。

[0011]

【発明の実施の形態】 以下、図1～図4を用いて本発明の実施の形態について説明する。図1は、本発明の異常検出方法によってCPUの異常を検出する制御用コンピュータ10の構成図である。この制御用コンピュータ10は、外部機器12から順次与えられる入力信号に基づいて演算処理を行う。そして、制御用コンピュータ10は、演算処理の結果においた制御信号を外部機器14に与えて外部機器14の動作制御を行なう。制御用コンピュータ10は、入力信号を供給する外部機器12は、例えば、スイッチやセンサ等である。また、制御用コンピュータ10によって制御される外部機器14は、例えば、アクチュエータやLED表示器等である。

[0012] 図1に示すように、制御用コンピュータ10は、入力ポート16、CPU18a、18b、メモリ20a、20b、受信バッファ22a、22b及び出力ポート24等を有する。入力ポート16は、スイッチやセンサ等で構成される外部機器12からの入力信号を順次取り込む。そして、入力ポート16は、ノイズ除去処理やレベルシフト処理等を行った後の入力信号をCPU18a、18bに与える。

[0013] CPU18aは、外部機器12から入力ポート16を介して与えられる入力信号に基づき演算処理を行なう。入力信号に応じた演算値a1、a2、a3、...を算出する。同様にCPU18bも外部機器12から入力ポート16を介して与えられる入力信号に基づき演算処理を行なう。入力信号に応じた演算値b1、b2、b3、...を算出する。

[0014] CPU18a、18bは、同期がとられておらず、共に所定の間隔T（例えば、約6ms）で演算値の算出を繰り返す。従って、演算値a1、a2、a3、...と演算値b1、b2、b3、...は、例えば、b1、a1、b2、a2、b3、a3、...のような順で交互に算出される。CPU18aによって算出された演算値a1、a2、a3、...は、メモリ20

aと受信バッファ22bに順次格納される。また、CPU18bによって算出された演算値b1、b2、b3、...は、メモリ20bと受信バッファ22aに順次格納される。

[0015] また、CPU18aは、最新の3つの自己の演算値（例えば、a1、a2、a3）をメモリ20aから読み出し、最新のCPU18bの演算値（例えば、b3）を受信バッファ22aから読み出す。そして、CPU18aは、演算値a1、a2、a3と演算値b3を比較し、比較結果に応じた制御信号を出力ポート24を介してアクチュエータやLED表示器等で構成される外部機器14に与える。この際、外部機器14は、CPU18aから与えられた制御信号に従って動作する。

[0016] 一方、CPU18bは、最新の3つの自己の演算値（例えば、b1、b2、b3）をメモリ20bから読み出し、最新のCPU18aの演算値（例えば、a3）を受信バッファ22bから読み出す。そして、CPU18bは、演算値b1、b2、b3と演算値a3を比較し、比較の結果、必要に応じてCPU18aから外部機器14への制御信号の出力を停止にする。

[0017] メモリ20a、20bは、それぞれCPU18a、18bの演算処理の結果である演算値a1、a2、...、b1、b2、...の他、CPU18a、18bの動作プログラム等を格納する。次に、CPU18a、18bの動作説明用フローチャートを用いて行なう。図2は、CPU18aが実行するルーチンを示すフローチャートである。図2に示すルーチンは、その処理が終了する毎に繰り返し起動される。なお、メモリ20a内には、前回と同回のルーチンで取得された入力信号に基づいてCPU18aの演算処理の結果である演算値a1、a2が既に格納されているものとする。また、受信バッファ22aには、CPU18bの演算処理の結果である最新の演算値b3が既に格納されているものとする。

[0018] 図2に示すルーチンが起動されると、まず、ステップ100において、外部機器12からの入力信号が入力ポート16を介して取得される。このステップ100の処理が終了すると、次に、ステップ102の処理が実行される。ステップ102では、CPU18bによる演算処理の結果である演算値b3を受信バッファ22aから取得される。そして、次に、ステップ104の処理が実行される。

[0019] ステップ104では、ステップ100において取得された入力信号に基づいた所定の演算が実行され、その結果、演算値a3が算出される。そして、続くステップ106では、ステップ104で算出された演算値a3がメモリ20aに格納される。また、演算値a3は、受信バッファ22bにも与えられ、受信バッファ22b内に格納される。このステップ106の処理が終了すると、次に、ステップ108の処理が実行される。

ステップ204の処理が実行される。

[0025] ステップ204では、ステップ200において取得された入力信号に基づいた所定の演算が実行され、その結果、演算値b3が算出される。そして、次のステップ206では、ステップ204で算出された演算値b3がメモリ20bに格納される。また、演算値b3は、受信バッファ22aにも与えられ、受信バッファ22a内に格納される。このステップ206の処理が終了すると、次に、ステップ208の処理が実行される。

[0026] ステップ208では、メモリ20bに格納されていた最新の3つの演算値b1、b2、b3と、ステップ202で取得されたCPU18aの最新の演算値a3との大きさの比較が行なわれる。ここで、例えば、演算値b1、b2、b3のうちの最大値をbMAX、最小値をbMINとする。また、比較時の最小演算値の範囲を定める判定余裕値を β とする。この判定余裕値 β は、予め設定されているものとする。ステップ208では、演算値a3と演算値bMAX + β 及びbMIN - β との大小関係の比較が行なわれる。そして、この比較処理の後には、ステップ210の処理が実行される。

[0027] ステップ210では、ステップ208の比較結果に基づいた判断処理が実行される。ステップ208において、 $a3 > bMAX + \beta$ 、又は、 $a3 < bMIN - \beta$ が成立しなれば、ステップ210において、CPU18aの演算値a3は、CPU18bの演算値b1、b2、b3と大きく乖離しており、CPU18a、18bは共に正値であると判断される。そして、このステップ210の処理が終了すると、再び、ステップ200の処理が実行される。一方、ステップ208において、 $a3 > bMAX + \beta$ 、又は、 $a3 < bMIN - \beta$ が成立するならば、ステップ210において、CPU18aの演算値a3は、CPU18bの演算値b1、b2、b3と大きく乖離しており、CPU18a、18bは共に正値であると判断される。この場合、次に、ステップ212の処理が実行される。

[0028] ステップ212では、CPU18aに停止信号が与えられ、CPU18aから外部機器14への制御信号の出力が停止となる。そして、今回のルーチンは終了となる。上記のように、CPU18a、18bが互いの演算値を比較し合うので、CPU18a、18bが正常であることが容易に判断できる。また、本発明によれば、CPU18a、18bの同期をとる必要がないため、同期発生回路からCPU18a、18bに同期信号を与えなくてもよい。

[0029] ここで、CPU18a、18bが互いの演算値を任意のタイミングで取得して、比較処理を行なうようにすると、相手側CPU（例えば、CPU18a）と比べてCPU18bの最新の演算値が自己CPU18aの最新の演算値より先に算出された値のため、算出された値の判断が正確でない。例えば、CPU18bの最新の

の演算値b3がCPU18aの最新の演算値a3より先に算出された値なのか後に算出された値なのか判別できない。この場合、CPU18aの今回の演算処理で算出された演算値a2と、今回（最新）の演算値a3と、演算値a3の次に算出される演算値a4とを考慮して、上記ステップ108の比較処理における演算値の許容範囲を定める判定余裕値 α を大きめに設定する必要がある。【0030】しかし、本発明では、上記ステップ10、2、104及びステップ202、204に示すように、CPU18a、18bは、共に相手側CPUの最新の演算値を受信バッファ22a、22bを介して取得してから、自己の演算処理を行っている。このため、相手側CPUの最新の演算値は、常に自己CPUの最新の演算値よりも先に算出されたものであると確定できる。

【0031】また、相手側CPUが最新の演算値を算出した後、自己CPUがその演算値を受信バッファを介して取得するまでは所定の微小な時間 τ （ $\tau < t$ ）を要するので、本発明において、自己CPUが受信バッファを介して取得した相手側CPUの最新の演算値は、自己CPUの前々回の演算処理時から今回（最新）の演算処理時に間に算出されたものであると確定できる。

【0032】そこで、本発明のステップ108、208では、相手側CPUの最新の演算値と、自己CPUの前々回、前回及び今回の演算処理時の演算値である最新の3つの演算値とを比較する構成にしている。このように、本発明では、相手側CPUの最新の演算値と比較する自己CPUの演算値を最小数の3つに限定している。比較処理における演算値の許容範囲を定める判定余裕値 α 、 β をより小さく値に設定することができる。判定余裕値 α 、 β をより小さく値に設定することで、CPU18a、18bの異常の検出がより高精度に行われる。

【0033】ここで、CPU18a、18bの演算値の比較処理が不要な入力番号に対しては、CPU18a、18bが個別に演算処理を行うようにしてもよい。演算値の比較処理が不要な入力番号をCPU18a、18bのいずれか一方に分配することで、2つのCPU18a、18bは有効に利用することができ、なお、上記実施例は、CPU18a、18bが共にそれぞれ最新の演算値を相手側CPUに与える構成であるが、例えば、CPU18aをマスタCPUとし、CPU18bをスレーブCPUとしてもよい。この場合、マスタCPUであるCPU18aがスレーブCPUであるCPU18bを起動させる。そして、CPU18aによって起動したCPU18bが受信バッファに格納していた最新の演算値をCPU18aに与える。

【0034】図4は、CPU18a、18bが共に正常である時の演算値a1、a2、a3及びb3を示す図である。なお、演算値a1、a2、b3、a3にそれぞれ

対応する時刻t1、t2、t3、t4は、各演算値がCPU18a、18bによって算出された時刻を示す。また、演算値a1、a2、b3、a3にそれぞれ対応する値A、B、C、Dは、各演算値a1、a2、b1、a3の大きさを示す。

【0035】図4に示すように、CPU18bによって時刻t3に算出された最新の演算値b3の値Cは、CPU18aによって時刻t1に算出された演算値a1の値Dより大きく、時刻t4に算出された演算値a3の値より小さい。従って、この場合、図2に示したルーチンのステップ108において、CPU10aは、CPU18a、18bが共に正常であると判断して外部機器14を制御するための制御信号を出力する。

【0036】図5は、CPU18a、18bの少なくとも一方が異常である時の演算値a1、a2、a3及びb3を示す図である。なお、演算値a1、a2、b3、a3にそれぞれ対応する時刻t1、t2、t3、t4は、各演算値がCPU18a、18bによって算出された時刻を示す。また、演算値a1、a2、b3、a3にそれぞれ対応する値A、B、C、Dは、各演算値a1、a2、b1、a3の大きさを示す。

【0037】図5に示すように、CPU18bによって時刻t3に算出された最新の演算値b3の値Cは、CPU18aによって時刻t1に算出された演算値a1の値Dに判定余裕値 α を加えた値よりも大きい。従って、この場合、図2に示したルーチンのステップ108において、CPU10aは、CPU18a、18bの少なくとも一方が異常であると判断して、外部機器14を制御するための制御信号の出力を停止する。同様に、CPU10bがCPU18a、18bの少なくとも一方が異常であると判断する場合は、CPU18aから外部機器14への制御信号の出力を禁止する。

【0038】なお、制御用コンピュータ10内のCPUの数は2個に限らず、制御用コンピュータ10内に3個以上のCPUを設けて、互いの演算値の比較によってCPUの異常検出を行なうようにしてもよい。上記実施例において、図2のステップ100及び図3のステップ200の処理が特許請求の範囲に記載の第1のステップに相当し、図2のステップ102及び図3のステップ202の処理が特許請求の範囲に記載の第2のステップに相当し、図2のステップ104及び図3のステップ204の処理が特許請求の範囲に記載の第3のステップに相当する。また、図2のステップ108及び図3のステップ208の処理が特許請求の範囲に記載の第4のステップに相当し、図2のステップ110及び図3のステップ210の処理が特許請求の範囲に記載の第5のステップに相当する。

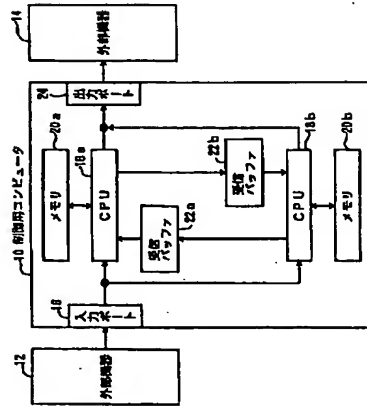
【0039】【発明の効果】 上述の如く、請求項1記載の発明によれば、複数のCPUを有するコンピュータシステムにおい

て、CPUの同期をとることなく、CPUの異常が容易に検出される。また、CPUの異常検出を高精度に行うことができる。また、請求項2記載の発明によれば、CPUの異常検出をより高精度に行うことができる。

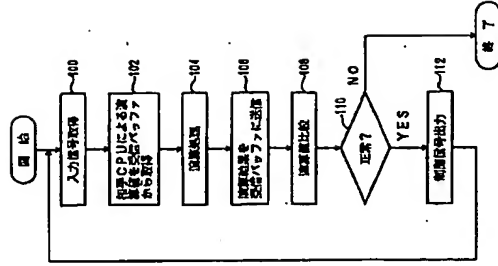
【図面の簡単な説明】
【図1】 本発明の異常検出方法が適用される制御用コンピュータの構成図である。
【図2】 CPUが実行するルーチンを示すフローチャートである。
【図3】 CPUが実行するルーチンを示すフローチャートである。
【図4】 2つのCPUが共に正常である時の演算値の値を示す図である。

【図1】 本発明の異常検出方法が適用される制御用コンピュータの構成図である。
【図2】 CPUが実行するルーチンを示すフローチャートである。
【図3】 CPUが実行するルーチンを示すフローチャートである。
【図4】 2つのCPUが共に正常である時の演算値の値を示す図である。

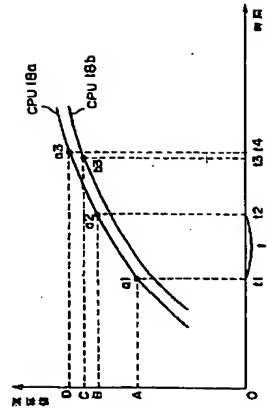
【図1】



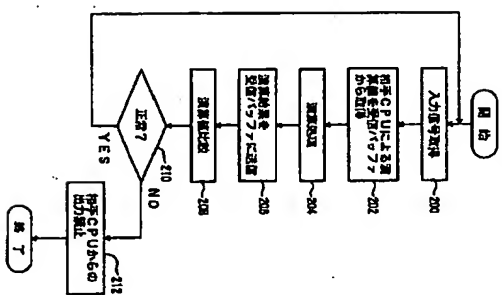
【図2】



【図4】



【図3】



【図5】

